BUNDESREPUBLIK DEUTSCHLAND

Prioritätsbescheinigung über die Einreichung einer Patentanmeldung

Aktenzeichen: 103 16 951.2

Anmeldetag: 12. April 2003

Anmelder/Inhaber: DaimlerChrysler AG, 70567 Stuttgart/DE

Bezeichnung: Verfahren zur Überprüfung der Datenintegrität von Software in Steuergeräten

IPC: G 06 F 9/445

Die angehefteten Stücke sind eine richtige und genaue Wiedergabe der ursprünglichen Unterlagen dieser Patentanmeldung.

München, den 04. März 2004
Deutsches Patent- und Markenamt
Der Präsident
Im Auftrag

A 9161
06/00
EDV-L

DaimlerChrysler AG

Eschbach
07.04.2003

Verfahren zur Überprüfung der Datenintegrität von Software in Steuergeräten

Die Erfindung betrifft ein Verfahren zum Aktualisieren und Laden von zumindest einem Anwenderprogramm, einer sogenannten Flashware, das in einem Programmspeicher eines Mikroprozessorsystems gespeichert werden soll. Der Downloadprozess erfolgt hierbei über eine Systemschnittstelle. Der Programmspeicher ist in einen elektrisch lösch- und programmierbaren Speicher, einen sogenannten Flash, und in einen flüchtigen Schreiblesespeicher, einem sogenannten Random Excess Memory, unterteilt. Bevor die herunterzuladende Flashware in dem Flashspeicher abgelegt wird, erfolgt eine Überprüfung der heruntergeladenen Programmdaten auf Integrität und Authentizität.

Ein Verfahren zum Aktualisieren und Laden von Anwenderprogrammen in einem Programmspeicher eines Mikroprozessorsystems ist aus der deutschen Patentschrift DE 195 06 957 C2 bekannt. Hier wird über eine Systemschnittstelle eine Flashware in den Flashspeicher eines Mikroprozessorsystems eingelesen. Die Flashware wird hierbei zunächst in einem statischen Schreiblesespeicher, einem sogenannten Static Random Excess Memory (SRAM), zwischengespeichert und mittels eines zyklischen Blocksicherungsverfahrens auf Übertragungsfehler überprüft. Eine Überprüfung auf Authentizität des heruntergeladenen Flashwareprogramms findet hierbei nicht statt.

Andererseits ist aus der deutschen Offenlegungsschrift DE 100 08 974 A1 ein Signaturverfahren für die Authentizitätsprüfung einer Flashware für ein Steuergerät in einem Kraftfahrzeug bekannt. Bei diesem Verfahren wird die Flashware mit einer sogenannten elektronischen Unterschrift versehen. Zur Erstellung der elektronischen Unterschrift wird von der Flashware mittels der an sich bekannten Hash-Funktion ein sogenannter Hash-Code generiert. Dieser Hash-Code wird mittels eines Public-Key-Verfahrens verschlüsselt. Als Public-Key-Verfahren wird vorzugsweise das RSA-Verfahren, benannt nach den Erfindern Rivest, Shamir und Adleman, eingesetzt. Der verschlüsselte Hash-Code wird dem zu übertragenden Anwendungsprogramm angehängt. Im Steuergerät wird der verschlüsselte Hash-Code mit dem öffentlichen Schlüssel entschlüsselt und mit dem im Steuergerät berechneten Hash-Code über die Flashware verglichen. Stimmen beide Hash-Codes überein, ist die übertragene Flashware authentisch. Eine Überprüfung auf Übertragungsfehler ist dem Signaturverfahren nicht zu entnehmen.

Ausgehend von dem vorbeschriebenen Stand der Technik ist es Aufgabe dieser Erfindung, ein Verfahren zur Überprüfung der Datenintegrität von Software in Steuergeräten vorzuschlagen, bei dem die übertragenen Daten in möglichst effizienter Weise auf Übertragungsfehler und Authentizität überprüft werden können.

Die erfindungsgemäße Lösung gelingt mit einem Verfahren mit den Merkmalen des unabhängigen Anspruchs. Vorteilhafte Ausgestaltungen des erfindungsgemäßen Verfahrens sind in den Unteransprüchen und in der Beschreibung der Ausführungsbeispiele enthalten.

Bei einer Überprüfung der Datenintegrität von Software bei einem Downloadprozess auf Übertragungsfehler und Authentizität müssen die geflashten Daten mehrmals überprüft werden. Der Zugriff bzw. die Zugriffszeit auf Programmdaten, die im Flashspeicher abgelegt sind, ist zeitintensiv. Besonders bei

Steuergeräten im Kraftfahrzeug, die aus Kostengründen in der Regel über geringe Rechenleistungen verfügen, führt eine lange Zugriffszeit bei aufwendigen Berechnungen, wie einer Authentizitätsprüfung, zu langen und unerträglichen Verzögerungen. Erfindungsgemäß kann die Überprüfung von Programmdaten auf Übertragungsfehler und Authentizität effizient gestaltet werden, wenn die Berechnungsverfahren zur Überprüfung auf Übertragungsfehler und für die Überprüfung auf Authentizität durchgeführt werden, solange sich die Flashware in einem Pufferspeicher mit schneller Zugriffszeit befindet. Zeitintensive Zugriffe auf den Flashspeicher werden dadurch vermieden. Musste bisher für jede Überprüfung der Flashware auf den Flashspeicher zugegriffen werden, so muss nach dem erfindungsgemäßen Verfahren lediglich einmal auf den Flashspeicher zugegriffen werden, um die Flashware für alle notwendigen Überprüfungen in einen Pufferspeicher mit schneller Zugriffszeit zwischenzuspeichern.

Der mit der Erfindung hauptsächlich erzielte Vorteil liegt in der zeitlich effizienten Berechnung von mehreren Prüfsummen und ggf. einer zusätzlichen Signaturprüfung durch Reduzierung der Zugriffe auf den Flashspeicher. Dies ermöglicht kürzere Flashzeiten für den Downloadprozess und damit etliche Einsparungen an Produktionszeit.

Für die Authentizitätsprüfung werden vorteilhafterweise an und für sich selbst bekannte Verfahren eingesetzt. Etablierte Standards sind z. B. die RSA-Signatur von Flashware oder die Verwendung eines sogenannten Message Authentication Code. Beide vorgenannten Authentizitätsprüfungen können mit Vorteil im Zusammenhang mit der Erfindung eingesetzt werden.

In einer alternativen Ausführung des erfindungsgemäßen Verfahrens erfolgt vor der Authentizitätsprüfung eine Abfrage und eine Auswahl der für die Authentizitätsprüfung anzuwendenden Sicherheitsklasse. Damit ist die Erfindung sowohl für

Flashware mit einer niederen Sicherheitsklasse als auch für Flashware mit einer hohen Sicherheitsklasse einsetzbar.

Im Folgenden wird die Erfindung anhand der Ausführungsbeispiele gemäß der Figuren 1 bis 3 näher erläutert.

Es zeigen:

Fig. 1 ein Blockdiagramm eines beispielhaften Steuergerätes mit einem Mikroprozessor und einer logisch funktionellen Aufteilung des Speicherbereichs.

Fig. 2 eine exemplarische Aufteilung eines Speichers in logische Blöcke, wobei jeder logische Block aus mehreren Segmenten bestehen kann. Die programmierten Daten (Flashware) werden in den Segmenten abgelegt. Die Lücken zwischen den Segmenten werden mit sogenanntem illegal opcode oder illegal data aufgefüllt.

Fig. 3 ein Ablaufdiagramm für das erfindungsgemäße Verfahren.

Figur 1 zeigt ein typisches Mikroprozessorsystem, wie es auch in Steuergeräten von Kraftfahrzeugen Verwendung findet. An einem Prozessorbus PBUS ist ein Mikroprozessor CPU, ein Systemspeicher sowie eine Systemschnittstelle Interface für die Kommunikation mit externen Systemen angeschlossen. Der Systemspeicher ist logisch und funktionell in verschiedene Speicherbereiche aufgeteilt. Diese Speicherbereiche können sowohl physikalisch voneinander getrennt sein als auch durch rein logische Segmentierung in einem physikalisch einheitlichen Speicher gebildet werden. In dem Boot-Sektor des Mikroprozessorsystems ist im Wesentlichen das Betriebssystem für den Mikroprozessor selbst abgelegt. Als Anwendungsprogramm ist in dem Boot-Sektor auch ein sogenannter Flash Boot Loader abgelegt. Mit diesem Flash Boot Loader werden bei Bedarf neue Anwendungsprogramme unter Systemschnittstelle Interface heruntergeladen und in den Flashspeicher des Mikroprozessorsystems

abgelegt. Weiterhin ist im Boot-Sektor die Hash-Funktion, nämlich der sogenannte RIPEMD-160-Algorithmus, abgespeichert. Im Flashspeicher Flash des Mikroprozessorsystems sind typischerweise die Anwendungsprogramme, mit denen das Steuergerät ECU arbeitet, abgelegt. Der Flashspeicher ist ein elektrisch löschbarer und programmierbarer, nicht flüchtiger Speicher. Derartige Speicher sind als EEPROM bekannt. Für die Anwendung des erfindungsgemäßen Verfahrens enthält das Mikroprozessorsystem einen Pufferspeicher Puffer. Dieser Pufferspeicher kann als separater Speicher, z. B. als sogenannter Cash-Speicher, ausgebildet sein oder kann als reservierter Speicherbereich innerhalb des Schreiblesespeichers RAM des Mikroprozessorsystems ausgebildet sein. In dem Schreiblesespeicher RAM werden von den Anwendungsprogrammen die notwendigen Daten, Zwischenergebnisse und Ergebnisse eingelesen, abgelegt, zwischengespeichert und ausgegeben. Für die Zwecke der Authentizitätsprüfungen ist in einem besonders geschützten Lesespeicher entweder ein Schlüssel in Form eines Dechiffriercodes oder in Form eines geheimen Kennzeichnungscodes hinterlegt. Ein Dechiffriercode wird für Verschlüsselungsverfahren benötigt, während ein Kennzeichnungscode für vereinfachte Authentifizierungsverfahren, wie z. B. die Message Authentication Codes, benötigt wird. Mit einem derartig aufgebauten Mikroprozessorsystem können Anwendungsprogramme als sogenannte Flashware mit einem Downloadprozess, wie er beispielsweise in der deutschen Patentschrift DE 195 06 957 C2 beschrieben ist, heruntergeladen werden und in dem Flashspeicher abgelegt werden. Auch ist es mit einem Mikroprozessorsystem gemäß dem Aufbau nach Figur 1 möglich, für die herunterzuladende Flashware standardisierte Authentifizierungsverfahren durchzuführen. Als Authentifizierungsverfahren im Sinne dieser Erfindung werden zum einen etablierte Signaturverfahren, wie z. B. die Public-Key-Verschlüsselung, bezeichnet und zum anderen die sogenannten Message Authentication Codes ins Auge ge-

fasst. Ein Beispiel eines Signaturverfahrens für Flashware, basierend auf einem Public-Key-Verfahren, ist ausführlich in der deutschen Patentanmeldung DE 100 08 974 A1 offenbart.

Bei den Public-Key-Verschlüsselungsverfahren hat sich das sogenannte RSA-Verschlüsselungsverfahren, benannt nach den Erfindern Rivest, Shamir und Adleman, als Standard durchgesetzt. Bei diesem Verfahren wird von der zu versendenden Nachricht zunächst ein Hash-Wert mit einer an sich bekannten Hash-Funktion, z. B. der Funktion RIPEMD-160, generiert. Der Sender verschlüsselt diesen berechneten Hash-Wert mit einem privaten und geheimen Schlüssel. Der verschlüsselte Hash-Wert bildet die Signatur und wird an die zu versendende Nachricht angehängt. Der Empfänger einer Nachricht entschlüsselt mit einem öffentlichen Schlüssel die Signatur und erhält dadurch wieder den vom Sender berechneten Hash-Wert. Weiter berechnet der Empfänger der Nachricht von der unverschlüsselten Originalnachricht mit der gleichen Hash-Funktion wie der Sender den Hash-Wert der Nachricht. Stimmen der Hash-Wert aus der entschlüsselten Signatur mit dem Hash-Wert, berechnet über die unverschlüsselte Nachricht, miteinander überein, ist die Nachricht integer und authentisch. Public-Key-Verschlüsselungsverfahren erfüllen hohe Sicherheitsanforderungen an Datenintegrität und Authentizität. In Bezug auf Steuergeräte in Kraftfahrzeugen und den Downloadprozess von Flashware für diese Steuergeräte erfüllen Public-Key-Verfahren die Bedingungen für diese höchste Sicherheitsklasse für den Downloadprozess der Flashware.

Allerdings sind Public-Key-Verschlüsselungsverfahren aufgrund der aufwendigen Verschlüsselungs- und Entschlüsselungsalgorithmen aufwendig und nicht auf jedem Mikroprozessor in einem Steuergerät eines Kraftfahrzeuges einsetzbar. Beispielsweise arbeiten die Verschlüsselungsverfahren mit Gleitkommaoperati-

onen, die von Mikroprozessoren in einfachen Steuergeräten nicht immer unterstützt werden. Authentifizierungsverfahren geringerer Sicherheitsstufe kommen ohne Chiffrierung und Dechiffrierung aus. Ein solches Verfahren hat sich als sogenannter Message Authentication Code MAC durchgesetzt. Ein Message Authentication Code arbeitet mit einem geheimen Identifizierungscode, den alle Kommunikationsteilnehmer kennen und haben müssen. Dieser Authentifizierungscode wird an die unverschlüsselte Nachricht angehängt und von der dermaßen gekennzeichneten Nachricht wird mittels einer Hash-Funktion ein Hash-Wert berechnet. Zwischen den Kommunikationsteilnehmern wird dann die unverschlüsselte Nachricht und der berechnete Hash-Wert ausgetauscht. Ein Empfänger überprüft die übermittelte Nachricht, indem er seinen Identifizierungscode an die unverschlüsselte Nachricht anhängt und hiervon, mit der gleichen Hash-Funktion wie der Sender, den Hash-Wert berechnet. Stimmen dieser berechnete Hash-Wert mit dem vom Sender übermittelten Hash-Wert überein, so gilt die empfangene Nachricht als integer und authentisch. Die Authentifizierungsverfahren, auf der Basis der vorbeschriebenen Message Authentication Codes, haben den Vorteil, dass lediglich ein an sich bekanntes Verfahren zur Hash-Wertberechnung eingesetzt werden muss. Weitere Chiffrier- oder Dechiffrierschritte, wie z. B. eine RSA-Verschlüsselung, werden hierbei nicht benötigt. Hash-Wertfunktionen können auch auf einfachsten Mikroprozessoren ausgeführt werden. Die Anwendung von Message Authentication Codes ist z. B. durch die Patentschrift US 6,064,297 belegt. Allerdings wurden Message Authentication Codes bisher lediglich bei Internetanwendungen oder, wie im Fall der US-Patentschrift, in Computernetzwerken bekannt.

Figur 2 nimmt Bezug auf die physikalische Datenverteilung in einem logischen oder physikalischen Speicherbereich bzw. Speicherblock. In einem Speicherblock sind in der Regel nicht

alle Speicherplätze mit Daten belegt. In der Regel befinden sich die Nutzdaten in einem Speicher in verschiedenen Segmenten, in denen der Speicherbereich beschrieben wurde. Zwischen den einzelnen Segmenten Segment 1, Segment 2 bis Segment N, wie in Figur 2 dargestellt, werden die nicht mit Nutzdaten beschriebenen Speicherbereiche mit sogenanntem illegal opcode oder illegal data aufgefüllt. Der illegal opcode bedeutet beispielsweise ein Anfüllen der nicht mit Nutzdaten beschriebenen Speicherbereiche mit logischen Nullen. Zur Überprüfung von logischen Speicherblöcken und zur Überprüfung von Kopiervorgängen auf Übertragungsfehler wurden in der Informationstechnologie die zyklischen Blocksicherungsverfahren entwickelt. In der englischen Bezeichnung heißen diese zyklischen Blocksicherungsverfahren Cyclic Redundancy Check, kurz CRC. Hierbei handelt es sich um eine Methode zur Überprüfung von Übertragungsfehlern mittels einer Checksumme. Ein einfaches Beispiel einer Checksumme ist das Paritätsbit, das zu jedem 8 Byte, 16 Byte, 32 Byte, 64 Byte-langen Informationspaket als Checksumme berechnet wird und angehängt wird. Das Paritätsbit gibt hierbei Auskunft darüber, ob die Anzahl der logischen Einsen in dem Informationspaket gerade oder ungerade ist. Ein Kopiervorgang gilt dann als fehlerfrei, wenn sich die Checksumme Parität beim Kopiervorgang nicht geändert hat. Diese zyklischen Blocksicherungsverfahren werden sowohl als Checksumme über den gesamten logischen Speicherblock, d. h. Nutzdaten in den Segmenten plus aufgefüllte Lücken, berechnet als auch als Checksumme über die Nutzinformation in den Segmenten alleine. Die Checksumme über den gesamten logischen Block wird hier mit CRC_total, während die Checksumme über die Nutzdaten in den Segmenten hier mit CRC_written bezeichnet wird. Diese zyklischen Blocksicherungsverfahren zur Überprüfung des Kopiervorgangs an sich, werden auch beim Downloadprozess von Flashware in die Flashspeicher eines Steuergerätes in einem Kraftfahrzeug angewandt. Zyklische Blocksiche-

rungsverfahren benötigen ähnlich wie eine Hash-Funktion Zugriff auf die Nutzdaten, deren Kopiervorgang bzw. deren Hash-Wert berechnet werden soll. Jedoch wurden bisher die zyklischen Blocksicherungsverfahren völlig getrennt von den mittels eines Hash-Wertverfahrens arbeitenden Authentifizierungsverfahrens durchgeführt. Das heißt, es wurden erst die Blocksicherungsverfahren durchgeführt und abgeschlossen, bevor man einen Hash-Wert für ein Authentifizierungsverfahren berechnet hat. Dadurch waren in Vergangenheit jeweils Lesezugriffe auf den Flashspeicher für die Blocksicherungsverfahren einerseits als auch im nachfolgenden Identifizierungsverfahren für die Hash-Wertberechnung andererseits notwendig.

An diesem Punkt setzt die Erfindung an.

Figur 3 zeigt ein Beispiel für einen optimierten Downloadprozess von Flashware, bei dem neben zyklischen Blocksicherungsverfahren auch ein Authentifizierungsverfahren, basierend auf einer Hash-Wertberechnung durchgeführt wird. Die in den Flashspeicher heruntergeladene Flashware wird zunächst aus dem Flashspeicher ausgelesen (read flash) und in den Pufferspeicher (refill buffer) zwischengespeichert. Im nächsten Schritt wird mit einem zyklischen Blocksicherungsverfahren über die gesamten, im Pufferspeicher zwischengespeicherten und aus dem Flashspeicher kopierten Daten eine Checksumme über den gesamten Flashspeicher berechnet. Mit dieser Checksumme CRC_total kann später die Integrität des Flashspeichers geprüft werden. In einem nächsten Abfrageschritt (data within segment?) wird abgefragt, ob der ausgelesene Flashspeicher Nutzdaten enthielt. Sind keine Nutzdaten vorhanden, wird nicht sofort ein Fehler ausgegeben, sondern erst beim Vergleich der berechneten Checksummen CRC_written mit der beim Downloadprozess übermittelten Checksumme CRC_transmitted. Die

Checksumme CRC_total wird gespeichert und steht damit bei einem späteren Selbstcheck zur Verfügung.

Enthielt der ausgelesene Flashspeicher Nutzdaten, wird für diese Nutzdaten ein separates Blocksicherungsverfahren durchgeführt. Dieses Blocksicherungsverfahren für die Nutzdaten wird lediglich über diejenigen Speicherbereiche durchgeführt, in denen die Nutzdaten abgelegt sind. Die berechnete Checksumme CRC_written wird später mit der beim Downloadprozess übertragenen Checksumme für die Nutzdaten der Originalsoftware CRC_transmitted verglichen. Für einen ordnungsgemäßen Kopiervorgang während des Downloadprozesses müssen beide Checksummen übereinstimmen. Stimmen die Checksummen CRC_written und CRC_transmitted nicht überein, wird wiederum eine Fehlermeldung „Error in CRC Verification" ausgegeben. Sofern die Flashware keiner besonderen Sicherheitsklasse unterliegt, werden an der zwischengespeicherten Flashware keine weiteren Prüfungen mehr vorgenommen. Unterliegt die Flashware besonderen Sicherheitsklassen, so werden unmittelbar anschließend an die Berechnung des CRC_written, die für die Authentifizierung der Flashware notwendigen Hash-Wertberechnungen durchgeführt. Da sich die Flashware zu diesem Zeitpunkt noch im Pufferspeicher, der im Vergleich zum Flashspeicher deutlich kürzere Zugriffszeiten hat, befindet, können die Hash-Wertberechnungen über die Daten im Pufferspeicher durchgeführt werden, was zu einem deutlich zeiteffizienteren Ablauf des Verfahrens führt. Die Hash-Wertberechnungen bzw. die Durchführung der Authentifizierungsverfahren müssen natürlich entsprechend der jeweiligen Sicherheitsklasse der Flashware durchgeführt werden. Von besonderem Interesse hierbei sind, wie im Zusammenhang mit Figur 1 bereits ausgeführt, Public-Key-Verschlüsselungsverfahren, in Form eines sogenannten RSA-Verfahrens, für Flashware mit einer hohen Sicherheitsklasse

oder die angeführten Message Authentication Codes für Flashware mit einer geringeren Sicherheitsstufe.

Ist die Flashware mit einem Message Authentication Code gesichert, wird die unverschlüsselte Flashware mit dem geheimen Identifizierungscode konkateniert und über diese Kombination ein Hash-Wert HMAC berechnet. Dieser berechnete Hash-Wert HMAC wird mit dem, beim Downloadprozess übermittelten Hash-Wert HMAC_transmitted verglichen. Stimmen beide Werte überein, ist die Authentifizierung erfolgreich (Verification ok), stimmen die beiden Werte nicht überein, wird eine Fehlermeldung ausgegeben „Error in HMAC-Verification".

Unterliegt die Flashware einer höheren Sicherheitsstufe, z. B. einer Authentifizierung durch das im Zusammenhang mit Figur 1 diskutierte RSA-Verfahren, so wird mit den im Puffer zwischengespeicherten Daten das Authentifizierungsverfahren gemäß diesem RSA-Verfahren durchgeführt. In diesem Fall wird der codiert übertragene Hash-Wert der Originalsoftware mit dem öffentlichen Schlüssel des RSA-Verfahrens dechiffriert, so dass man den Hash-Wert der Originalsoftware Hash_transmitted erhält. Sodann wird für die im Pufferspeicher befindliche Flashware ein weiterer Hash-Wert Hash (CCC) berechnet und mit dem dechiffrierten Hash-Wert der Originalsoftware Hash_transmitted verglichen. Stimmen beide Hash-Werte überein, ist die Authentifizierung erfolgreich (Verification ok). Stimmen beide Hash-Werte nicht überein, wird eine Fehlermeldung ausgegeben „Error in Hash Verification". Gelingt eine Dechiffrierung des codiert übertragenen Hash-Wertes nicht, so endet das Authentifizierungsverfahren vorzeitig und es wird eine Fehlermeldung „Error in Signature Verification" ausgegeben.

Zusammenfassend kann festgehalten werden, dass durch die Zwischenspeicherung der heruntergeladenen Flashware in einem Pufferspeicher mit schnellen Zugriffszeiten die für den Downloadprozess notwendigen Prüfverfahren zeiteffizienter durchgeführt werden können. Sowohl die zyklischen Blocksicherungsverfahren als auch die je nach Sicherheitsklasse anzuwendenden Authentifizierungsverfahren werden in dem erfindungsgemäßen Verfahren mit den im Pufferspeicher zwischengespeicherten Daten durchgeführt. Ein mehrfacher Zugriff auf den Flashspeicher für die Durchführung der Blocksicherungsverfahren einerseits und für die Durchführung der Authentifizierungsverfahren andererseits wird erfolgreich vermieden. Dadurch ergeben sich letztlich kürzere Flashzeiten und damit eine Einsparung von Produktionszeit. Der Downloadprozess für Flashware muss nämlich bei einem Download in ein Steuergerät eines Kraftfahrzeuges zum ersten Mal während der Produktion des Kraftfahrzeuges durchgeführt werden. Die Kraftfahrzeuge können schließlich nicht mit Steuergeräten ohne Software ausgeliefert werden.

DaimlerChrysler AG

Eschbach
07.04.2003

Patentansprüche

1. Verfahren zur Überprüfung der Datenintegrität von Flashware in elektronischen Steuergeräten mit mindestens einem Mikroprozessor (CPU), mindestens einem Flashspeicher (Flash), mindestens einem Boot-Sektor, mindestens einem Pufferspeicher und mindestens einer Schnittstelle (Interface) für das Herunterladen der Flashware,
d a d u r c h g e k e n n z e i c h n e t ,
dass zur Überprüfung der Datenintegrität die Flashware in einen Pufferspeicher geladen wird und das für die Flashware im Pufferspeicher mindestens zwei Prüfsummen berechnet werden, nämlich ein zyklisches Blocksicherungsverfahren zur Überprüfung auf Übertragungsfehler und eine Hash-Wertberechnung zur Überprüfung der Flashware auf Authentizität.

2. Verfahren nach Anspruch 1,
d a d u r c h g e k e n n z e i c h n e t ,
dass für die Flashware im Pufferspeicher ein zyklisches Blocksicherungsverfahren (CRC) sowie eine Authentifizierung durch einen Message Authentication Code und eine Hash-Wertberechnung durchgeführt werden.

3. Verfahren nach Anspruch 1,
d a d u r c h g e k e n n z e i c h n e t ,
dass für die Software im Pufferspeicher ein zyklisches Blocksicherungsverfahren, eine Signaturprüfung und eine Hash-Wertberechnung durchgeführt werden.

4. Verfahren nach Anspruch 3,
d a d u r c h g e k e n n z e i c h n e t ,
dass die Signaturprüfung mit einem Public-Key-Verfahren erfolgt.

5. Verfahren nach einem der Ansprüche 1 bis 5,
d a d u r c h g e k e n n z e i c h n e t ,
dass nach dem Blocksicherungsverfahren eine Abfrage der Sicherheitsklasse für die zu überprüfende Software erfolgt.

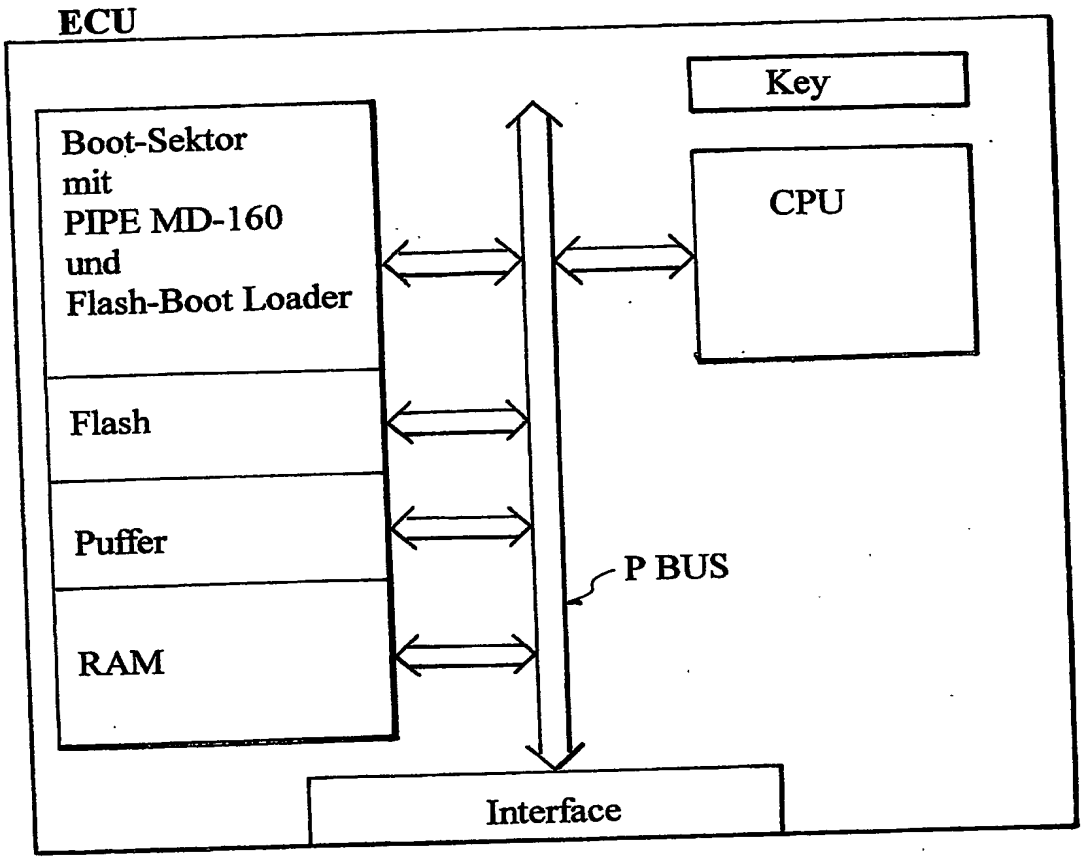
ECU
Key
Boot-Sektor
mit
PIPE MD-160
und
Flash-Boot Loader
CPU
Flash
Puffer
RAM
P BUS
Interface

FIG. 1

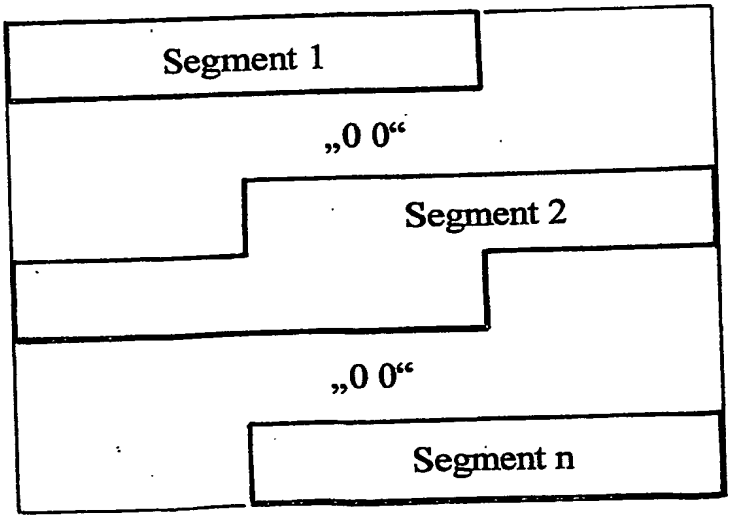
Segment 1
„0 0“
Segment 2
„0 0“
Segment n

FIG. 2

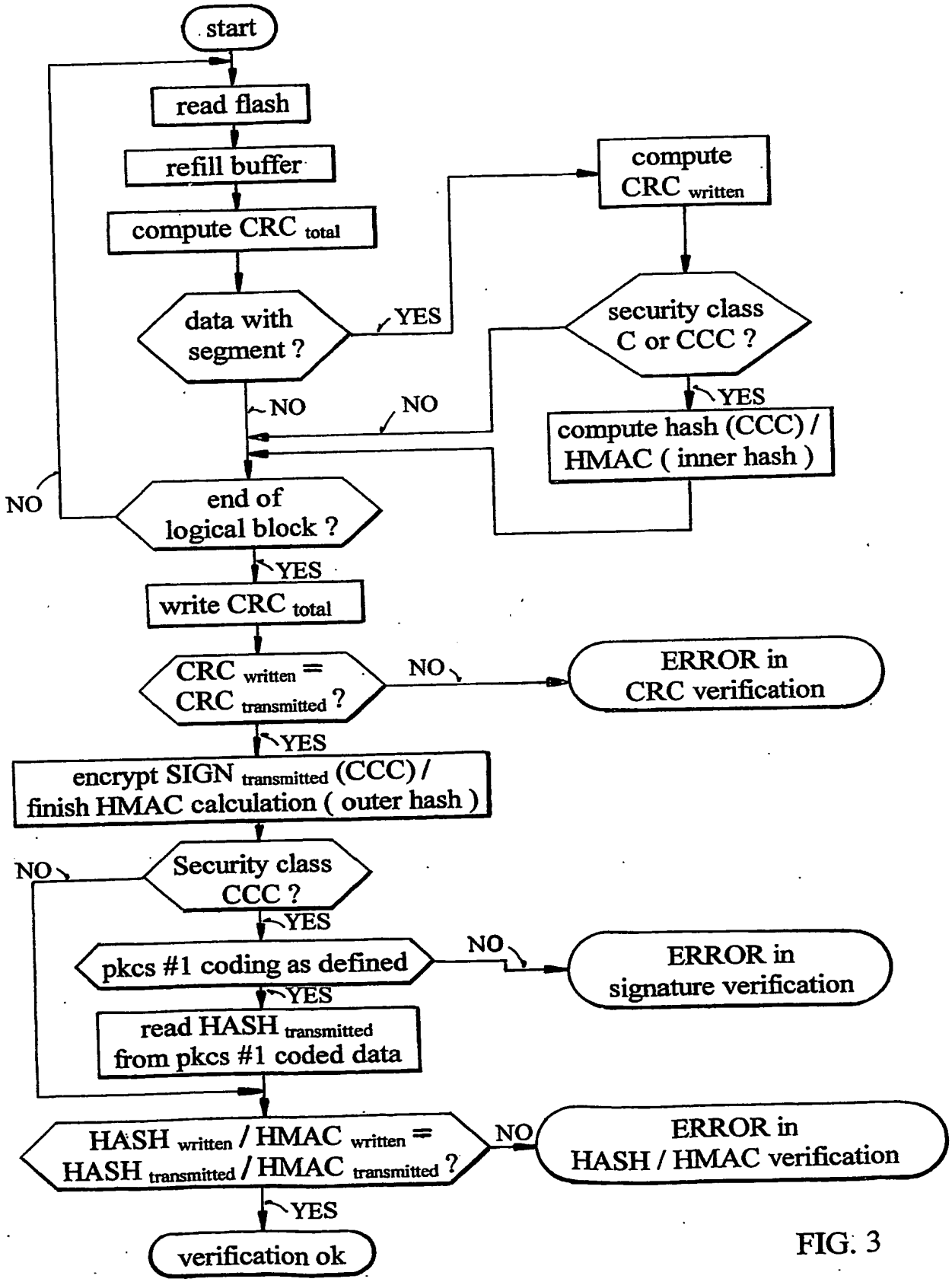
start
read flash
refill buffer
compute CRC total
data with segment ?
YES
NO
compute CRC written
security class C or CCC ?
YES
NO
compute hash (CCC) / HMAC (inner hash)
NO
end of logical block ?
YES
write CRC total
CRC written = CRC transmitted ?
NO
ERROR in CRC verification
YES
encrypt SIGN transmitted (CCC) / finish HMAC calculation (outer hash)
NO
Security class CCC ?
YES
pkcs #1 coding as defined
NO
ERROR in signature verification
YES
read HASH transmitted from pkcs #1 coded data
HASH written / HMAC written = HASH transmitted / HMAC transmitted ?
NO
ERROR in HASH / HMAC verification
YES
verification ok

FIG. 3

DaimlerChrysler AG

Eschbach
07.04.2003

Zusammenfassung

Bei einer Überprüfung der Datenintegrität von Software bei einem Downloadprozess auf Übertragungsfehler und Authentizität müssen die geflashten Daten mehrmals überprüft werden. Der Zugriff bzw. die Zugriffszeit auf Programmdaten, die im Flashspeicher abgelegt sind, ist zeitintensiv. Besonders bei Steuergeräten im Kraftfahrzeug, die aus Kostengründen in der Regel über geringe Rechenleistungen verfügen, führt eine lange Zugriffszeit bei aufwendigen Berechnungen, wie einer Authentizitätsprüfung, zu langen und unerträglichen Verzögerungen. Erfindungsgemäß kann die Überprüfung von Programmdaten auf Übertragungsfehler und Authentizität effizient gestaltet werden, wenn die Berechnungsverfahren zur Überprüfung auf Übertragungsfehler und für die Überprüfung auf Authentizität durchgeführt werden, solange sich die Flashware in einem Pufferspeicher mit schneller Zugriffszeit befindet. Zeitintensive Zugriffe auf den Flashspeicher werden dadurch vermieden. Musste bisher für jede Überprüfung der Flashware auf den Flashspeicher zugegriffen werden, so muss nach dem erfindungsgemäßen Verfahren lediglich einmal auf den Flashspeicher zugegriffen werden, um die Flashware für alle notwendigen Überprüfungen in einen Pufferspeicher mit schneller Zugriffszeit zwischenzuspeichern.

BEST AVAILABLE IMAGES

Defective images within this document are accurate representations of the original documents submitted by the applicant.

Defects in the images include but are not limited to the items checked:

- [x] BLACK BORDERS
- [x] IMAGE CUT OFF AT TOP, BOTTOM OR SIDES
- [x] FADED TEXT OR DRAWING
- [ ] BLURED OR ILLEGIBLE TEXT OR DRAWING
- [ ] SKEWED/SLANTED IMAGES
- [x] COLORED OR BLACK AND WHITE PHOTOGRAPHS
- [ ] GRAY SCALE DOCUMENTS
- [ ] LINES OR MARKS ON ORIGINAL DOCUMENT
- [ ] REPERENCE(S) OR EXHIBIT(S) SUBMITTED ARE POOR QUALITY
- [ ] OTHER: _____

IMAGES ARE BEST AVAILABLE COPY.
As rescanning documents *will not* correct images problems checked, please do not report the problems to the IFW Image Problem Mailbox